Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

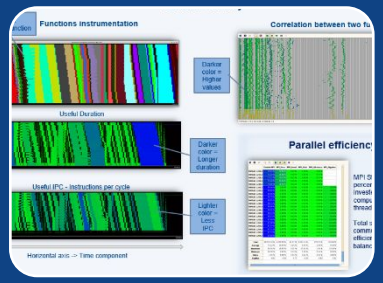
HARMONIE performance analysis: summary of the 1st phase

Xavier Yepes-Arbós
Mario C. Acosta

Who we are

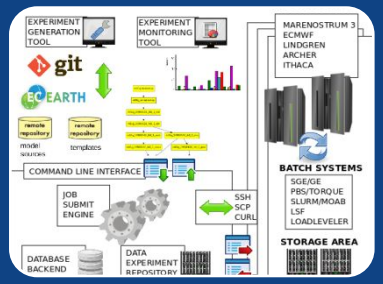
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Computational Earth Sciences Group



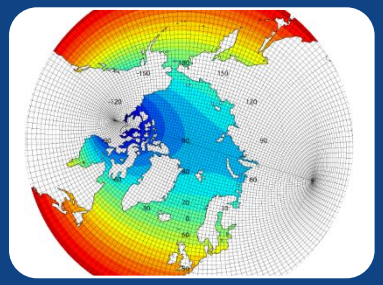
Performance Team

- Provide HPC Services (profiling, code audit, …)
- Apply new computational methods



Models and Workflows Team

- Development of HPC user-friendly software framework
- Support the development of atmospheric research software



Data and Diagnostics Team

- Big Data in Earth Sciences
- Provision of data services
- Visualization

Performance Team

- The necessary refactoring of numerical codes is given a lot of attention and is stirring a number of discussions
  - Computational performance analysis and new optimizations are needed for actual numerical models
  - Studying new algorithms for the new generation of high performance platforms (path to exascale)
- We are collaborating with several institutions on different projects working together in the same direction

is-enes
INFRASTRUCTURE FOR THE EUROPEAN NETWORK FOR EARTH SYSTEM MODELLING

immerse
IMPROVING OCEAN MODELS FOR THE COPERNICUS PROGRAMME

High Resolution Hirlam Limited Area Model

esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

MERCATOR OCEAN INTERNATIONAL

ETP 4 HPC EsDs

ESCAPE 2

PRACE

intel

BSC Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Roadmap

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

BSC-HIRLAM collaboration

- The BSC and the HIRLAM consortium signed a contract for a 1 year project to perform a complete code profiling of the HARMONIE-AROME model and the Data Assimilation system

- The project consists of two phases:
  - 1st: Basic profiling analysis
  - 2nd: Perform a complete analysis according to the results from the first phase

Scope of the phase 1

- Duration: 4 months
- Prepare selected configurations to be deployed with Extrae on cca/ccb at ECMWF, a Cray XC40 machine
- Perform a basic analysis of the HARMONIE-AROME Forecast model and the Data assimilation execution
  - Use different computational metrics: IPC, useful duration, MPI overhead, cache misses, etc
  - Identify the different parts of the trace with regard the code being executed
- Deliver a complete document with the results and feedback to decide the main goals for the profiling analysis of the phase 2

Scope of the phase 2

- Duration: 8 months after completion of phase 1
- Complete profiling analysis according to the results obtained from phase 1
- Training:
  - Prepare basic tutorials based on coarser HARMONIE configurations
  - Prepare a physical event to perform a training for the users
- Prepare complete documentation:
  - Online follow-up meetings to detect deviations and correct if needed
  - Write a final document describing all the tasks carried on
- Presence in the HIRLAM System group meetings: dissemination and feedback

Code profiling methodology

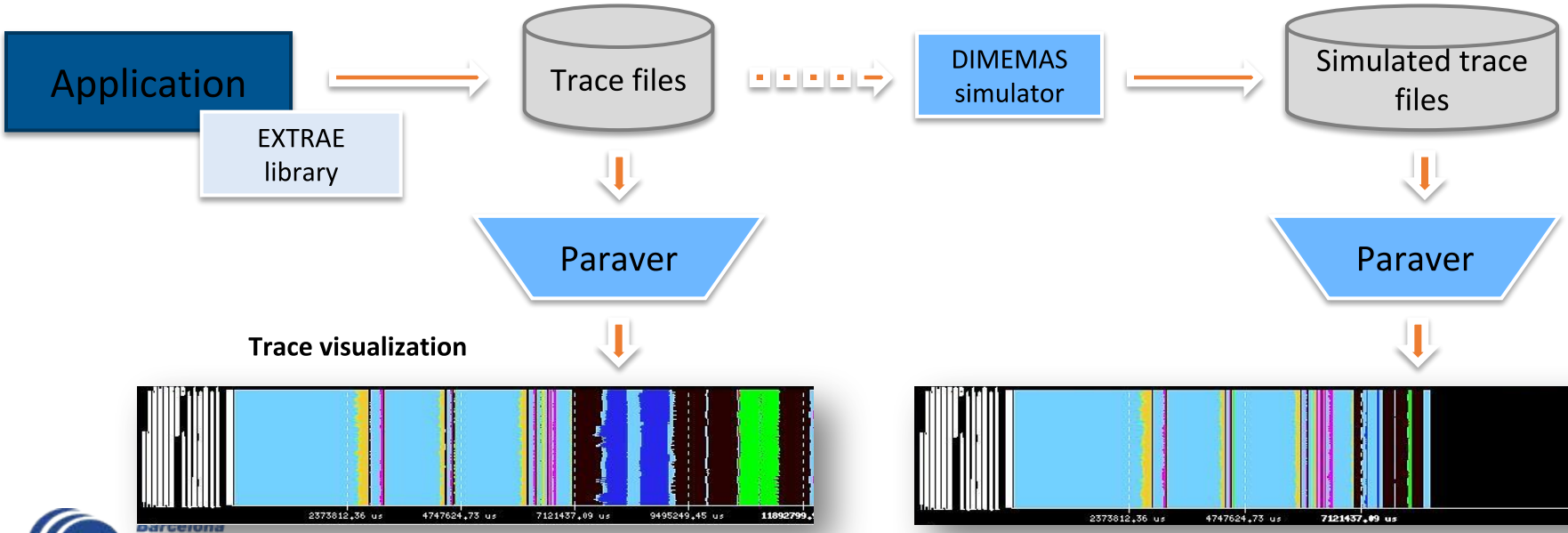
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Profiling methodology overview

- Scalability tests: MPI, OpenMP, Tiling
- Evaluate deployment efficiency: compilation flags
- Affinity tests: find a proper placement for MPI processes
- Profile analysis: user functions calls, statistics…
- Trace analysis: MPI, hardware counters, communication…
- Performance simulation: evaluate the code under machine changes
- Validation tests: check code correctness if optimized

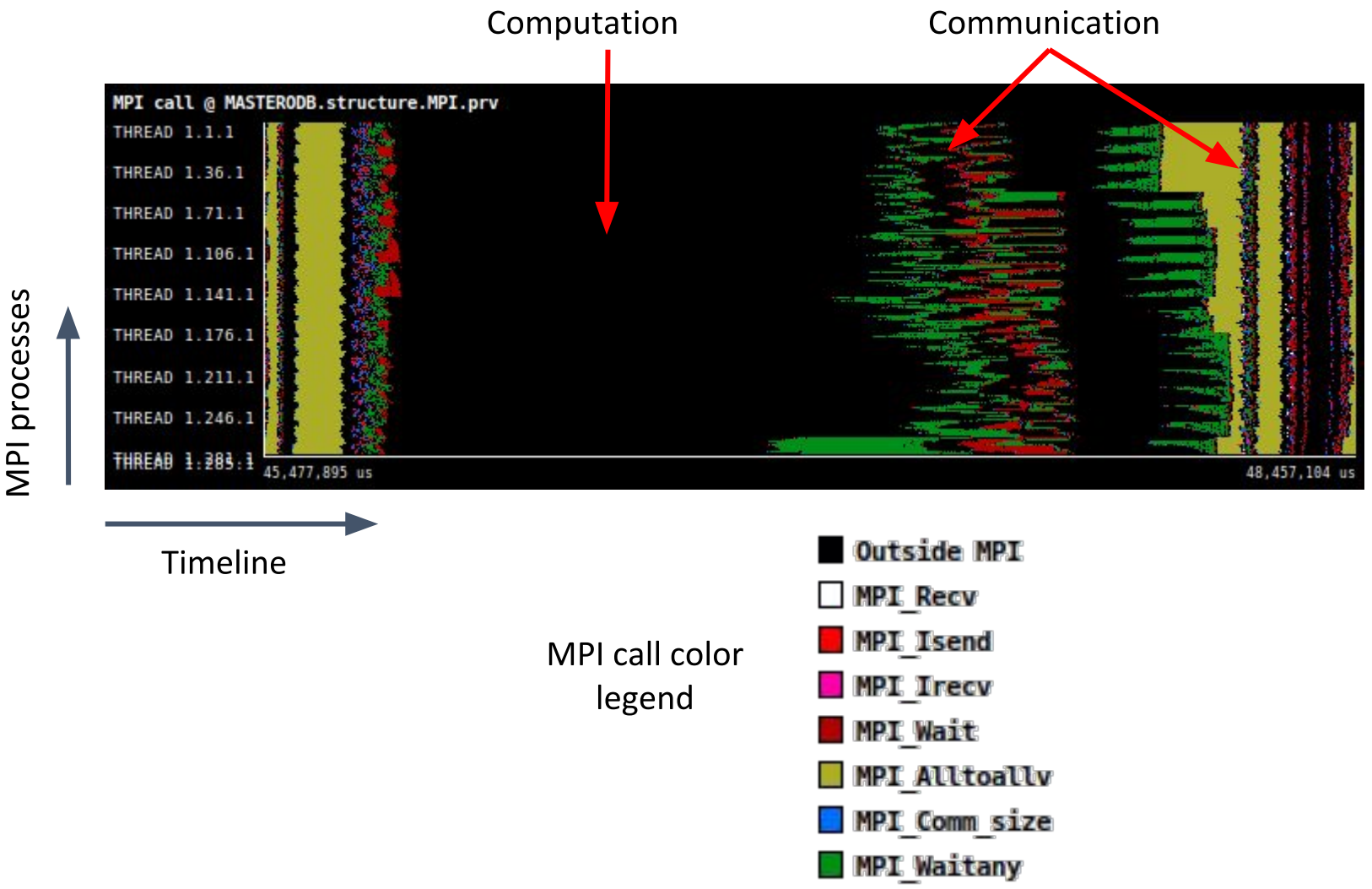
BSC performance tools

- Since 1991
- Based on traces
- Open Source: http://www.bsc.es/paraver
- **Extrae**: Package that generates Paraver trace-files for a post-mortem analysis
- **Paraver**: Trace visualization and analysis browser
  - Includes trace manipulation: Filter, cut traces
- **Dimemas**: Message passing simulator



DIMEMAS generated trace. Target = ideal machine

How a trace looks like: basic overview



Model configuration and structure

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

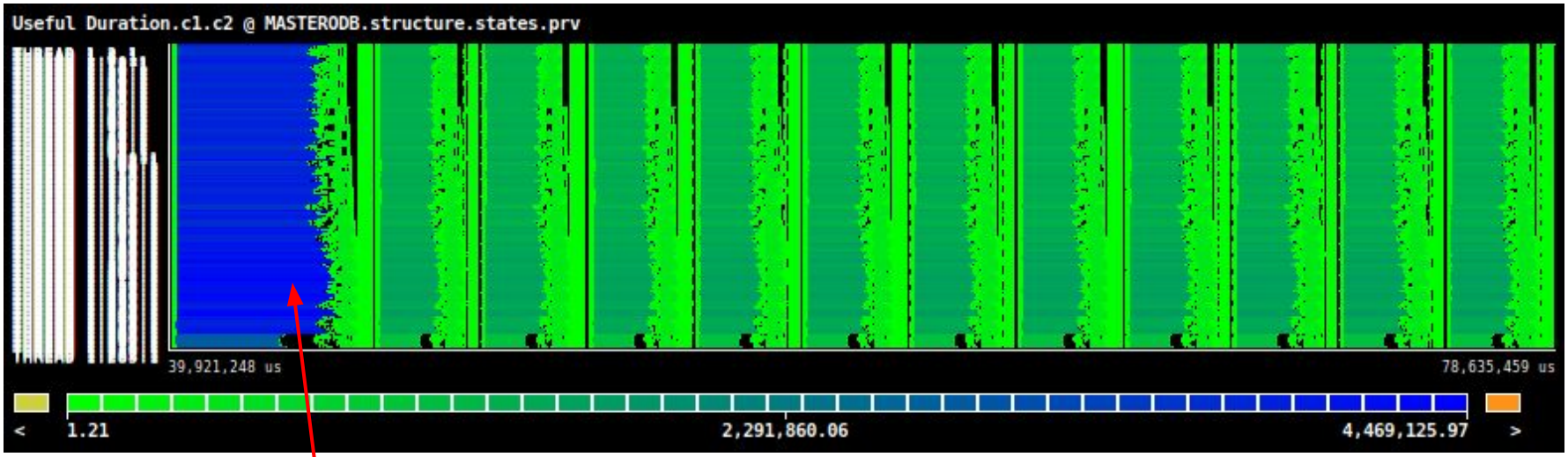
HARMONIE configuration

Common configuration for the different scenarios:

Branch	release-43h2.beta.5 (with OpenMP bugfix)
Domain	METCOOP25C
Number of points (x)	900
Number of points (y)	960
Grid size in meters (x, y)	2500.0
Time step	75 seconds
Forecast length	12 hours
Compile environment	gcc 7.3.0
MPI library	Cray MPICH 7.5.3
NPROMA size	30
Output	disabled

Structure of HARMONIE

Cut trace containing 12 time steps



Regular time step plus radiation

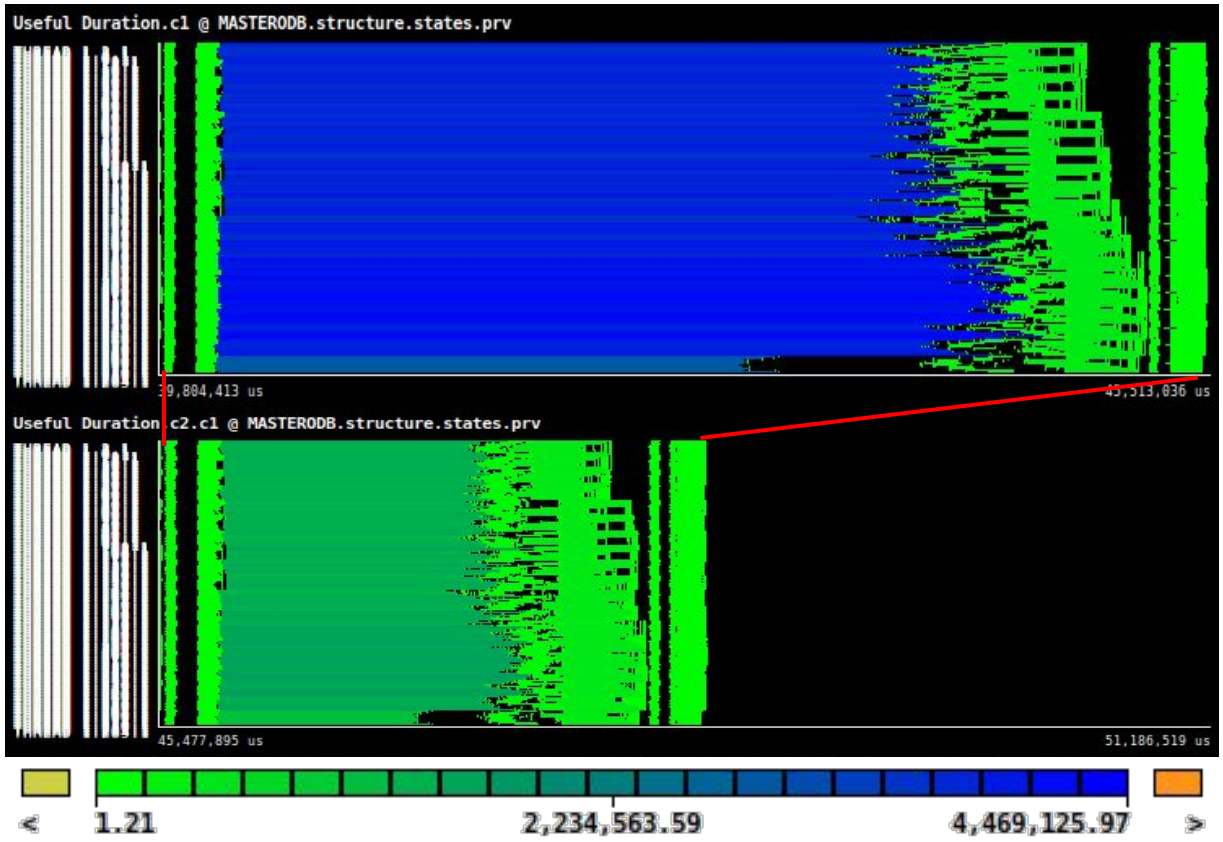
Regular time steps

Types of time step

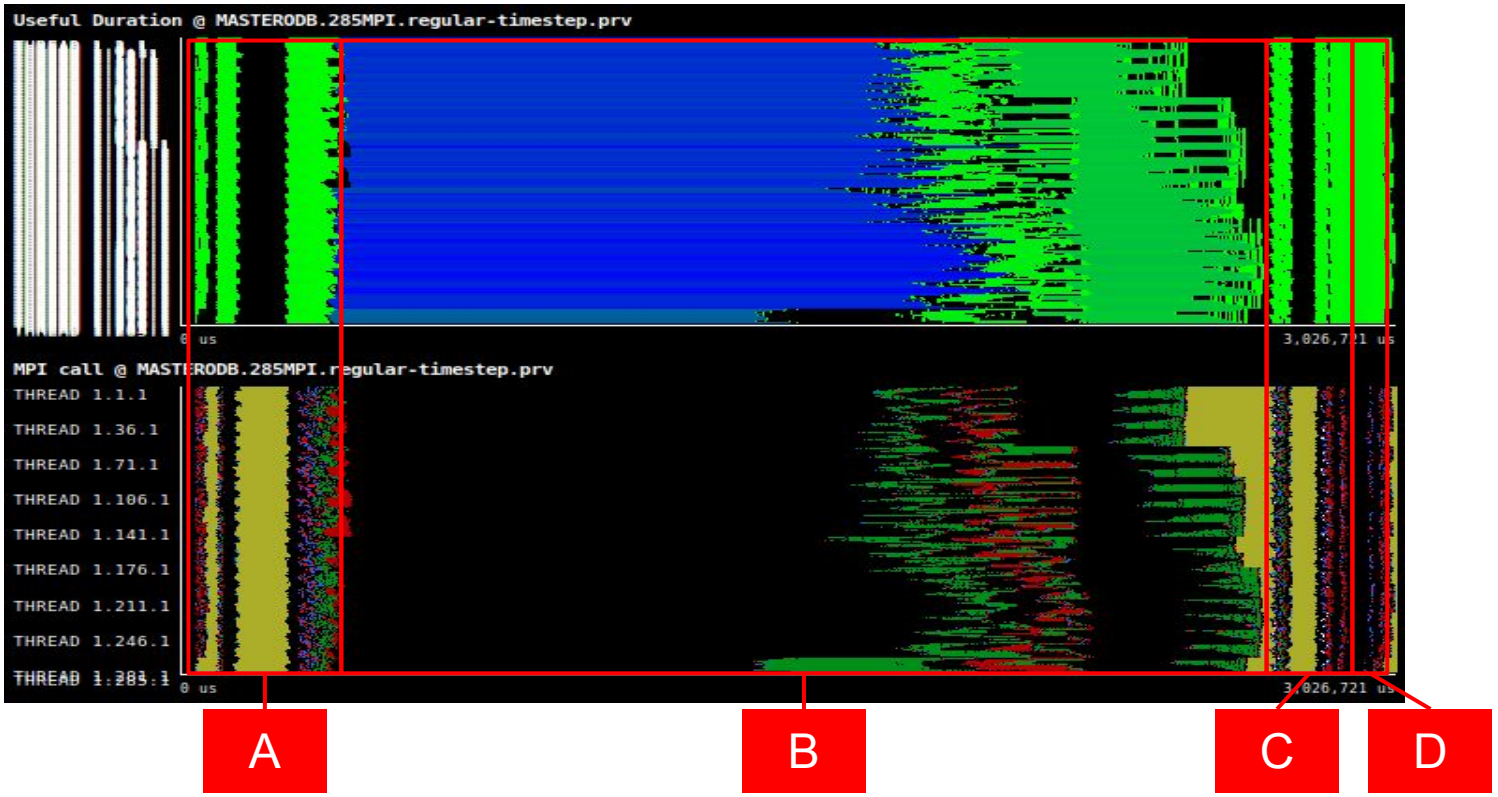
Time steps with radiation are much more expensive due to the extra computation in the grid-point part

Regular time step plus radiation

Regular time step



Structure of a regular time step



A - Inverse transformations
B - Grid-point computations
C - Direct transformations
D - Spectral computations

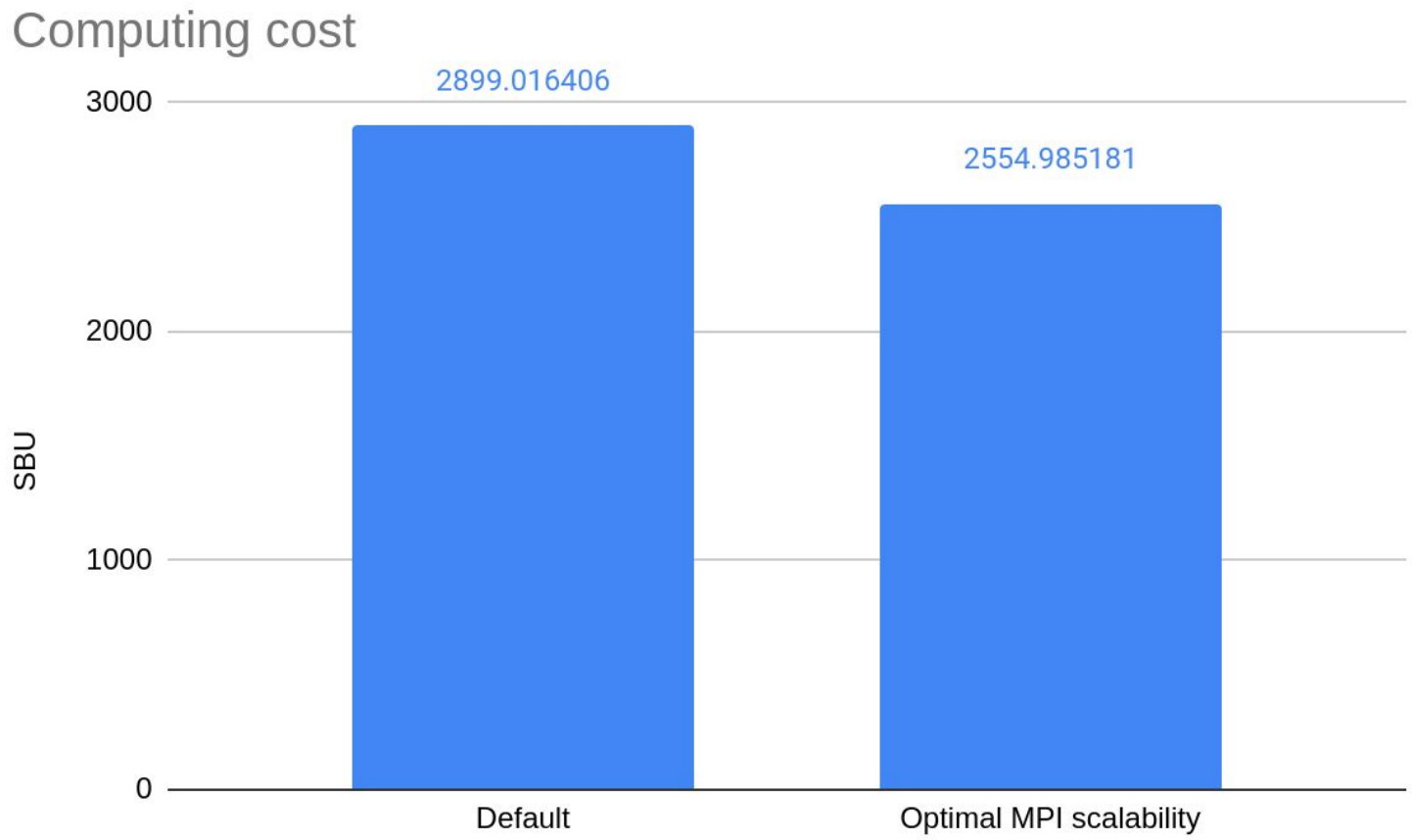
Deployment efficiency evaluation

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Potential optimizations for MPI only

- Compilation flags:
  - -ftree-vectorize: It **increases** the execution time
  - -march=native: It **decreases** the execution time
  - -O3: It **increases** the execution time
- OpenMP:
  - No OpenMP support: It **decreases** the execution time
- Job geometry:
  - -ss: the execution time is not affected, but it **might be decreased** in other circumstances
- MPI master in a shared node:
  - It **saves** computing resources

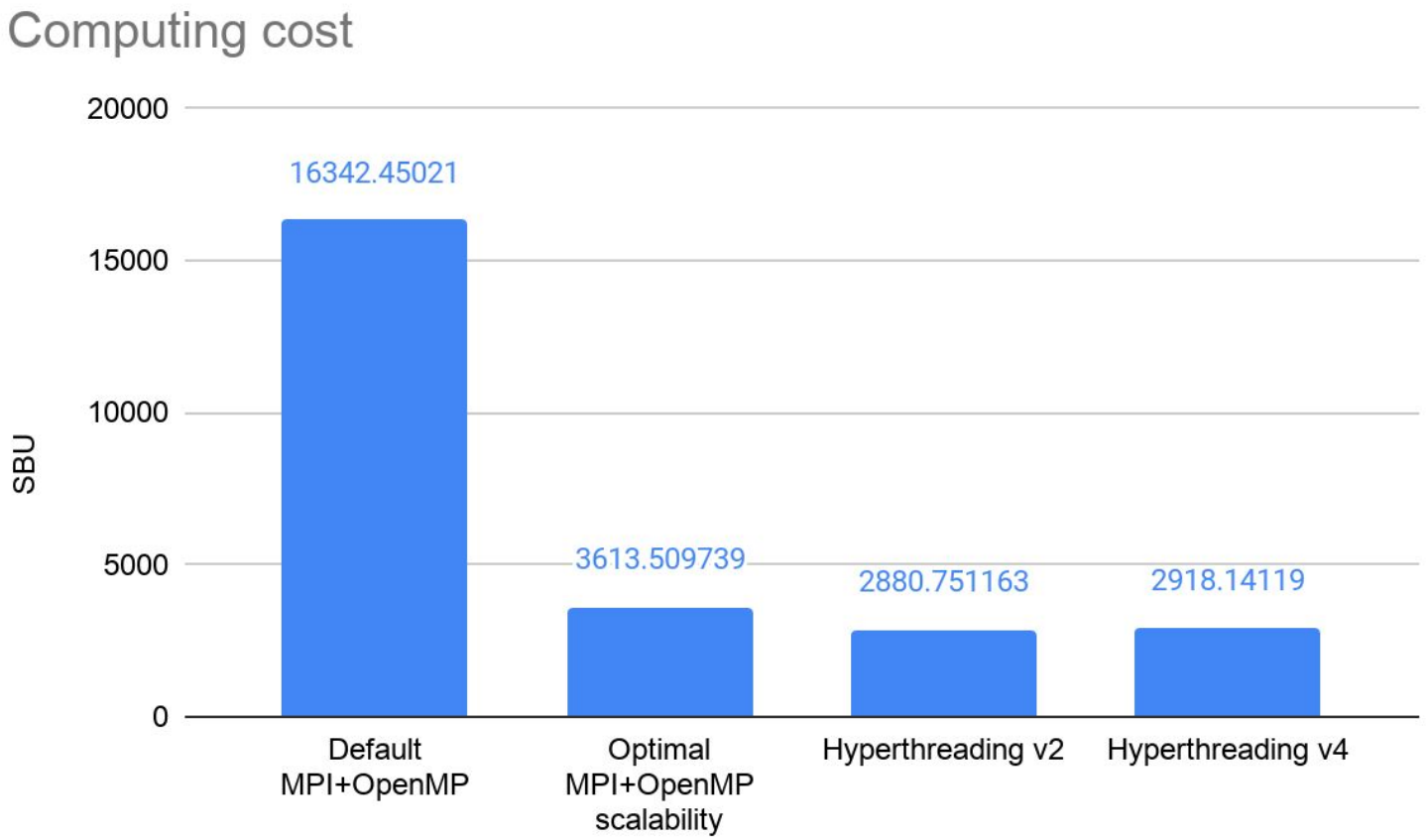
Computing cost of MPI only setups



Potential optimizations for MPI+OpenMP

- All previous useful MPI only optimizations, except that OpenMP is enabled
- Job geometry:
  - EC_threads_per_task and -d: It **decreases** the execution time
  - -cc depth: It **decreases** the execution time
  - Hyperthreading: depending on the needs (time-to-solution vs. cost-to-solution), it might be interesting to enable hyperthreading to **decrease** the execution time or **save** computing resources

Computing cost of MPI+OpenMP setups



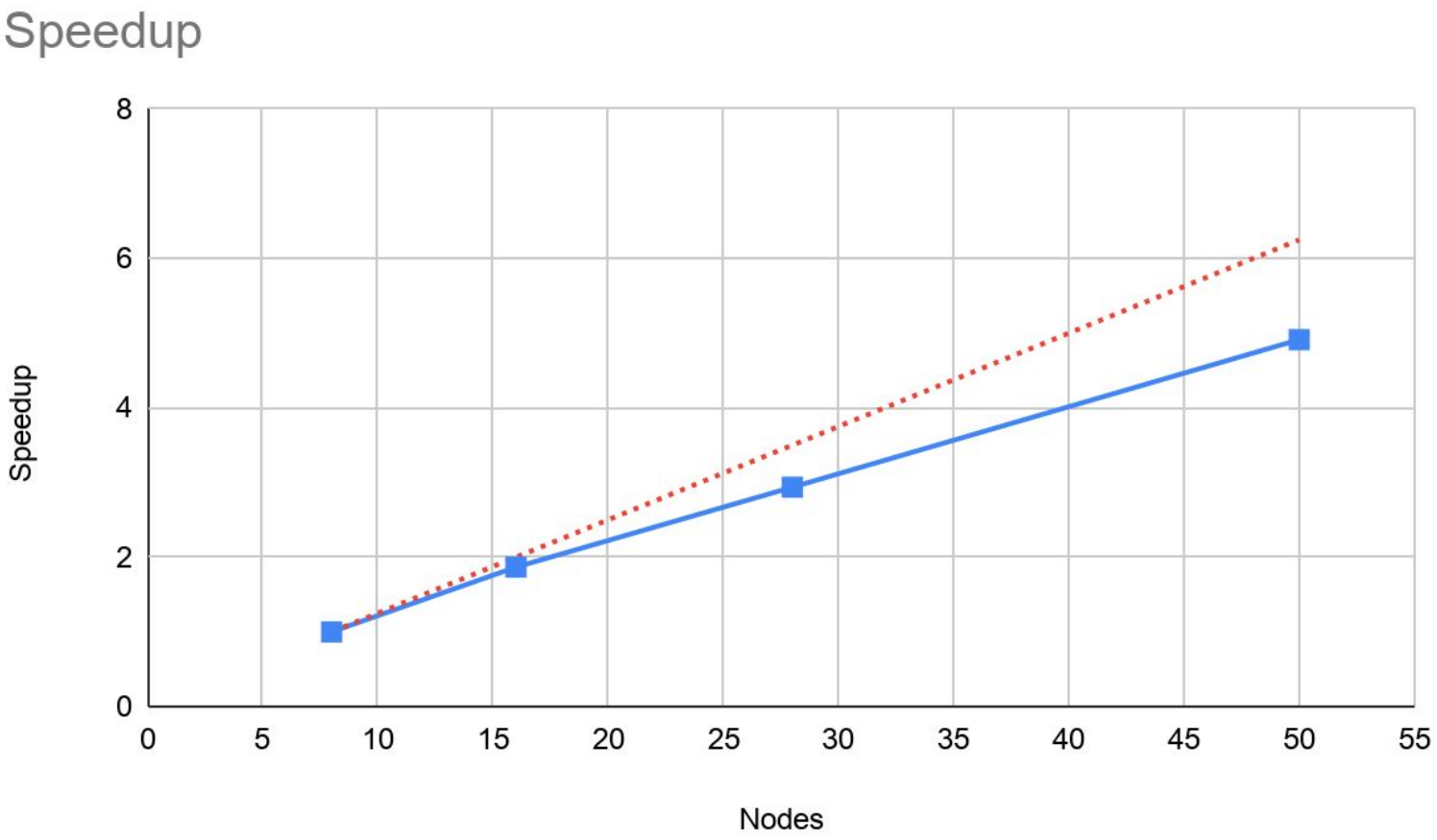
Scalability tests

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

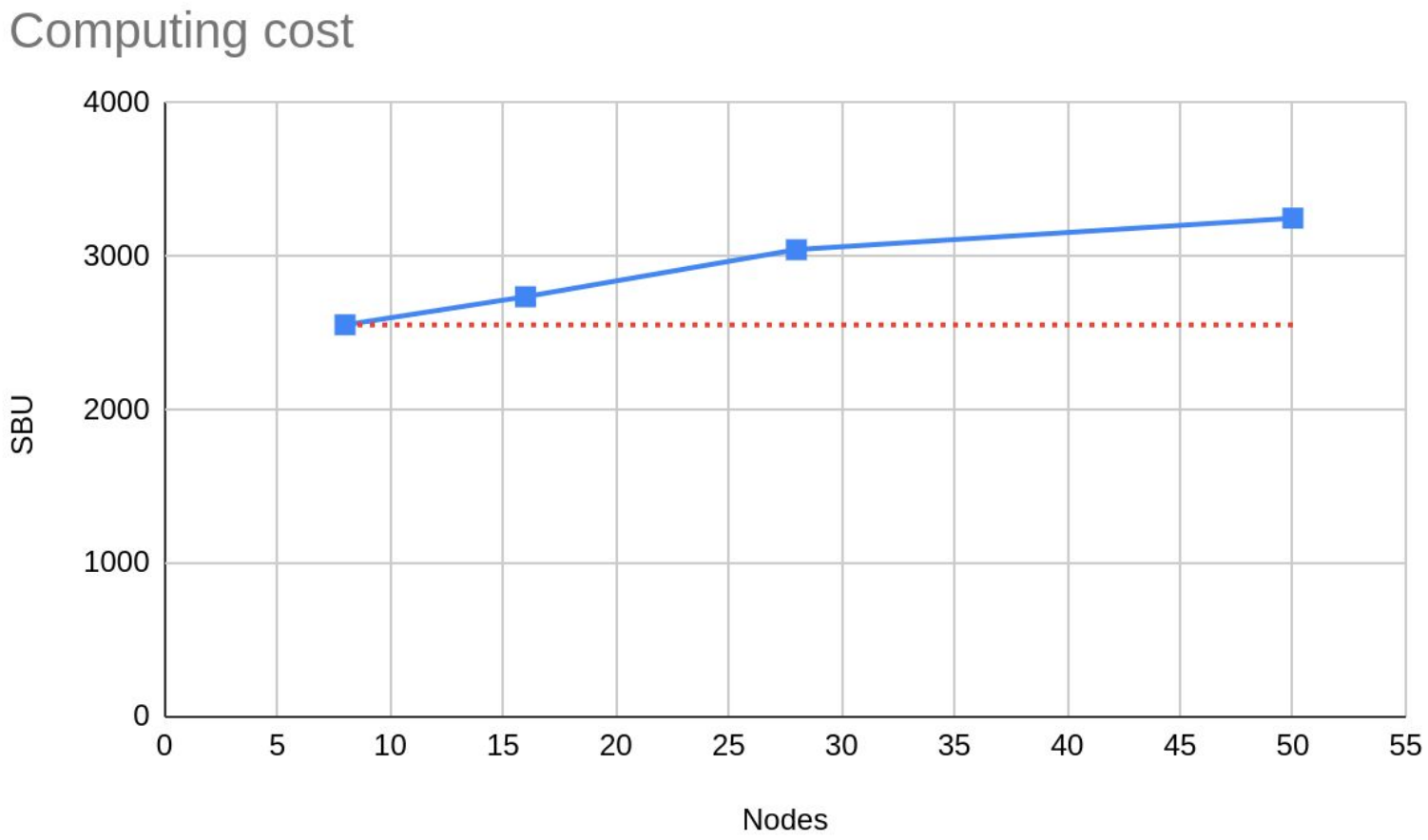
MPI strong scaling: test cases

MPI processes	nprocx	nprocy	Nodes
285	15	19	8
576	18	32	16
1008	28	36	28
1800	25	72	50

MPI strong scaling: speedup



MPI strong scaling: computing cost



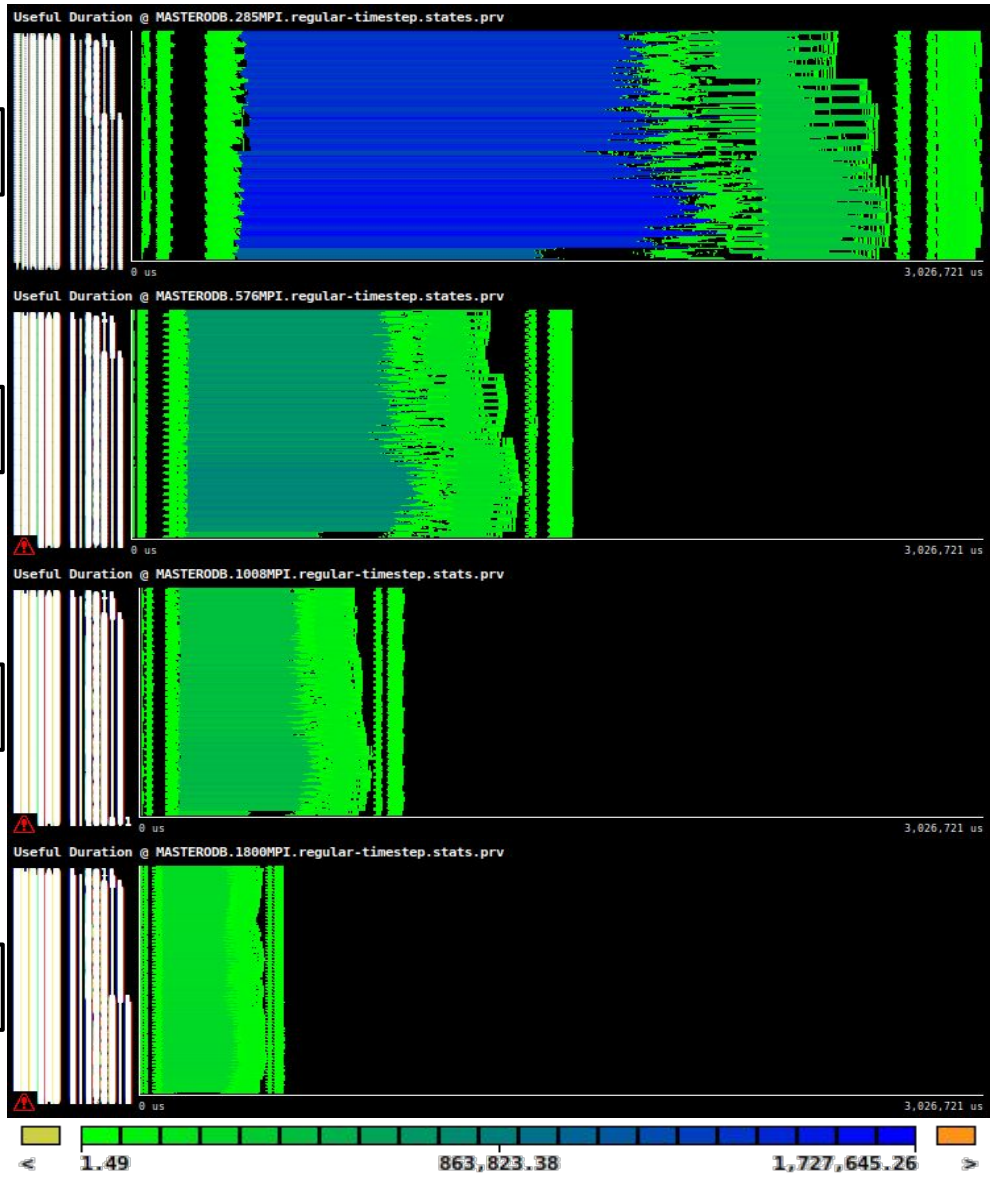
MPI strong scaling: trace views

285 MPI (8 nodes)

576 MPI (16 nodes)

1008 MPI (28 nodes)

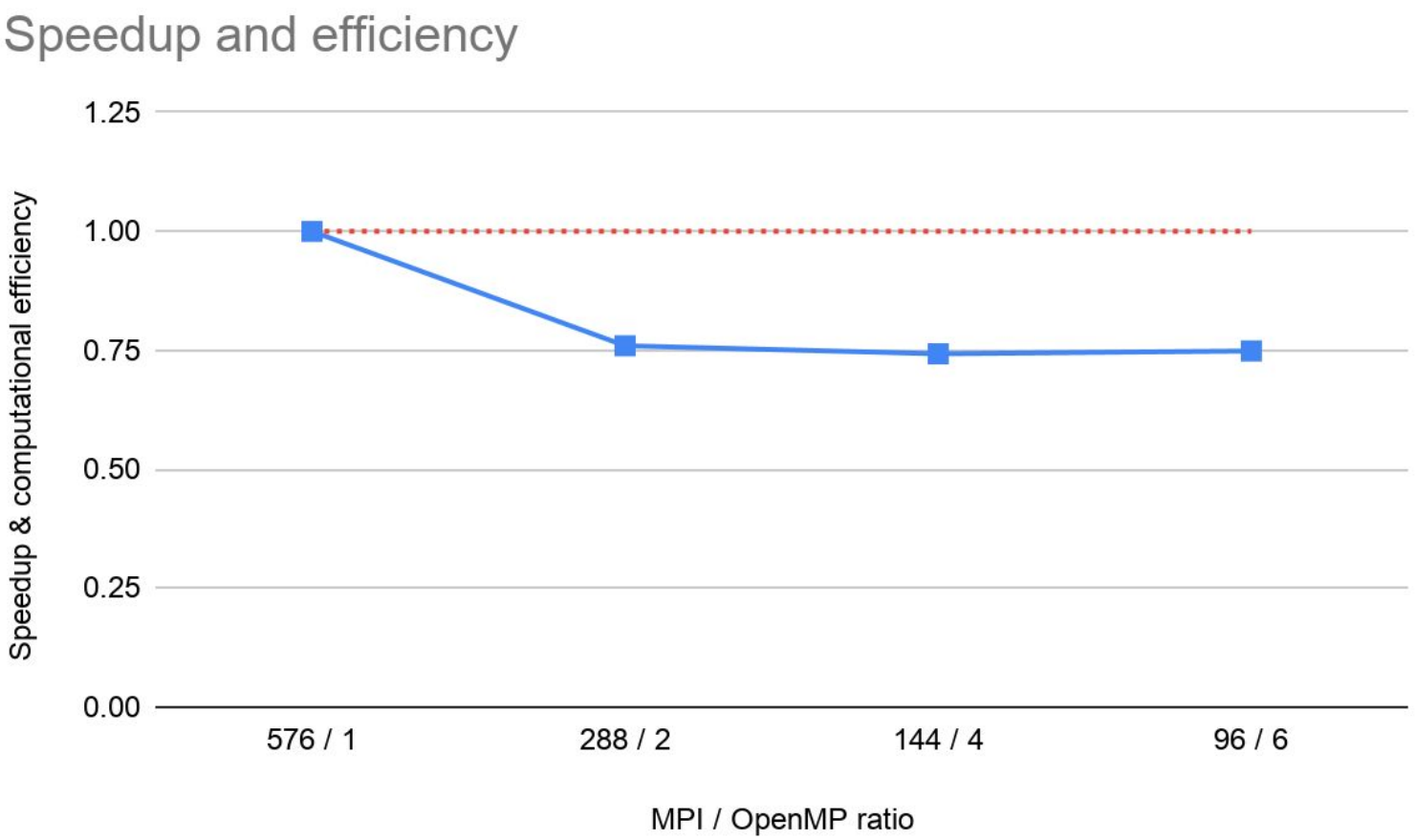
1800 MPI (50 nodes)



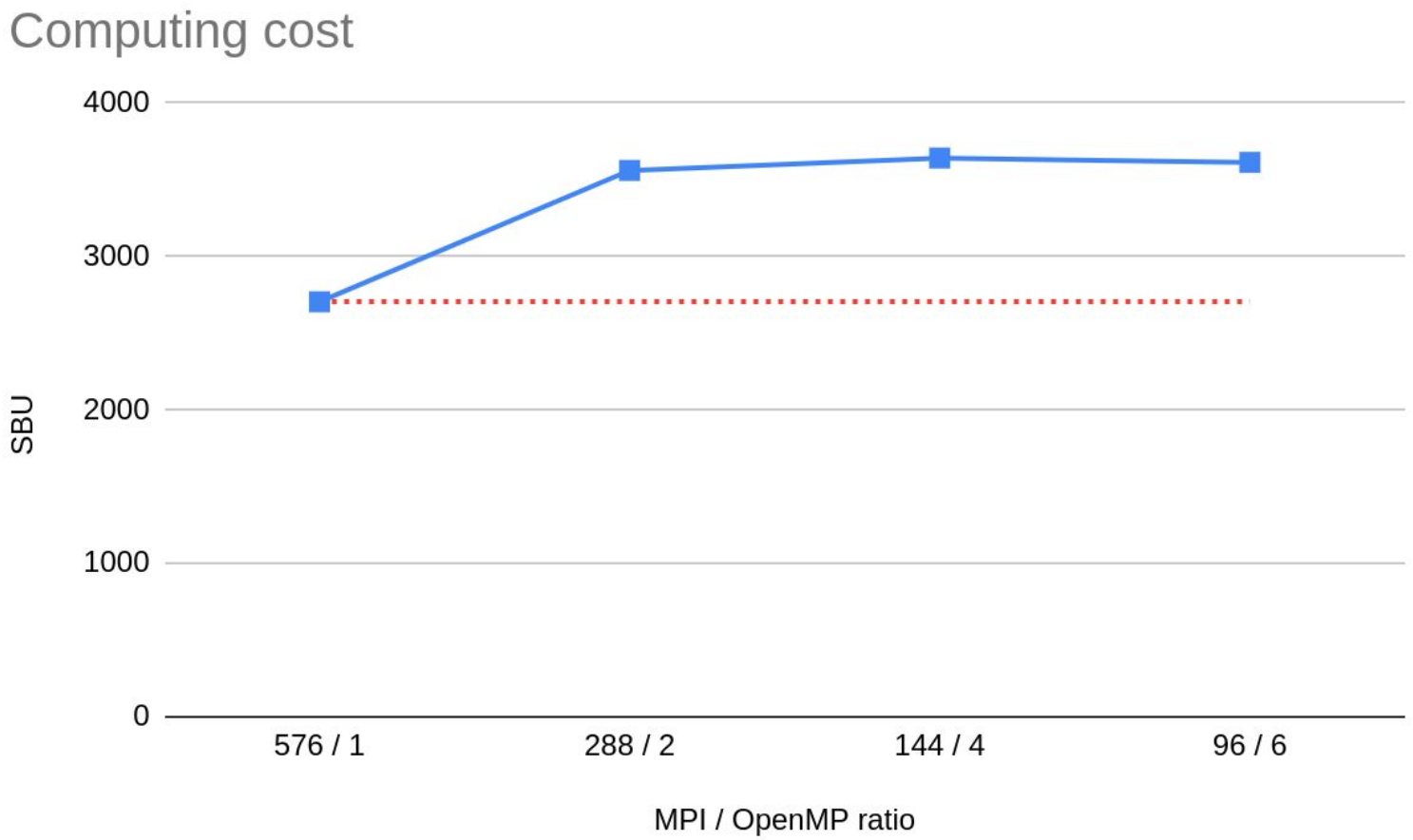
MPI+OpenMP scaling: test cases

OpenMP threads	MPI processes	nprocx	nprocy
1	576	18	32
2	288	16	18
4	144	9	16
6	96	8	12

MPI+OpenMP scaling: speedup



MPI+OpenMP scaling: computing cost



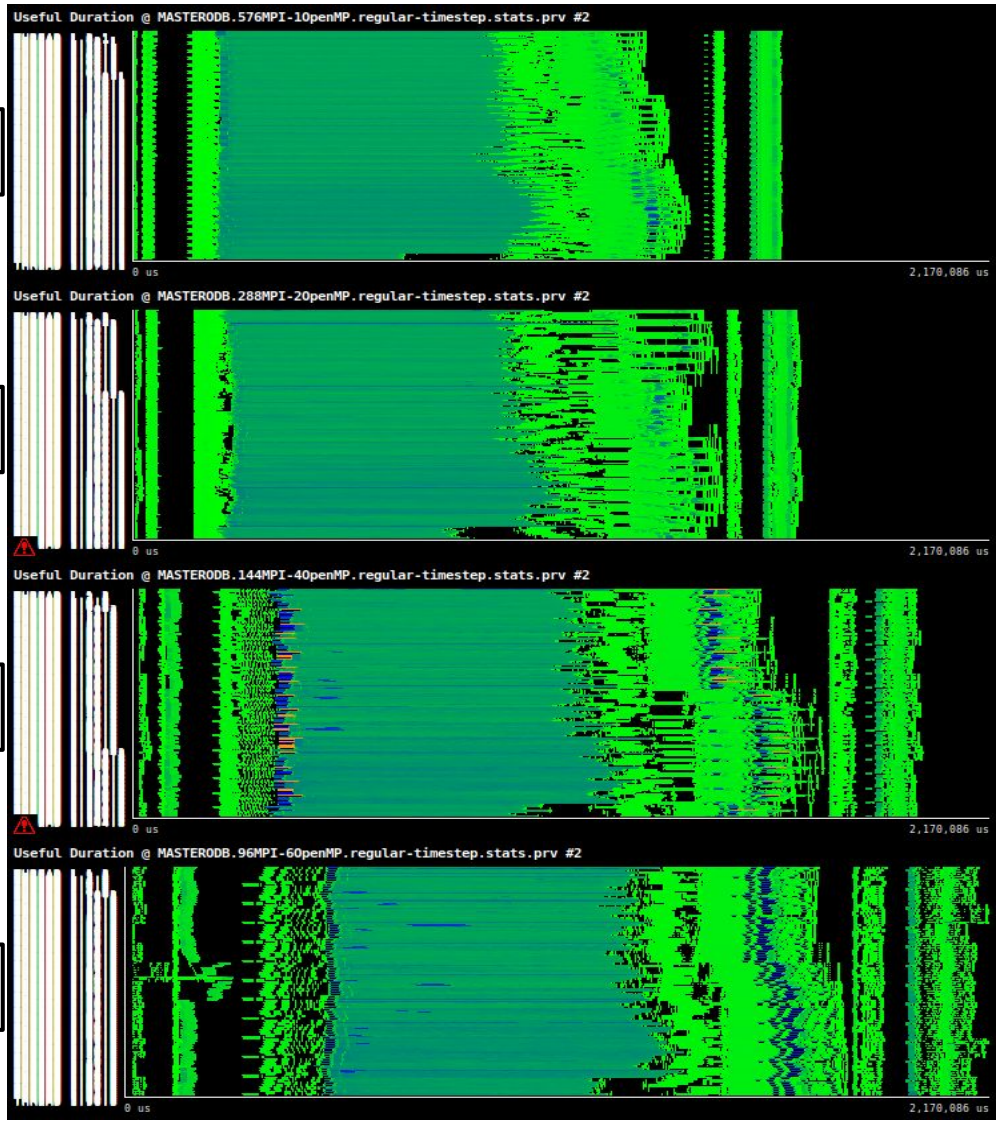
MPI+OpenMP scaling: trace views

576 MPI / 1 OMP

288 MPI / 2 OMP

144 MPI / 4 OMP

96 MPI / 6 OMP



Profiling analysis

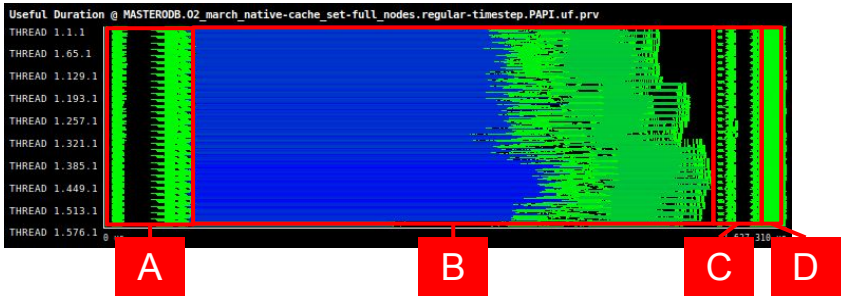
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

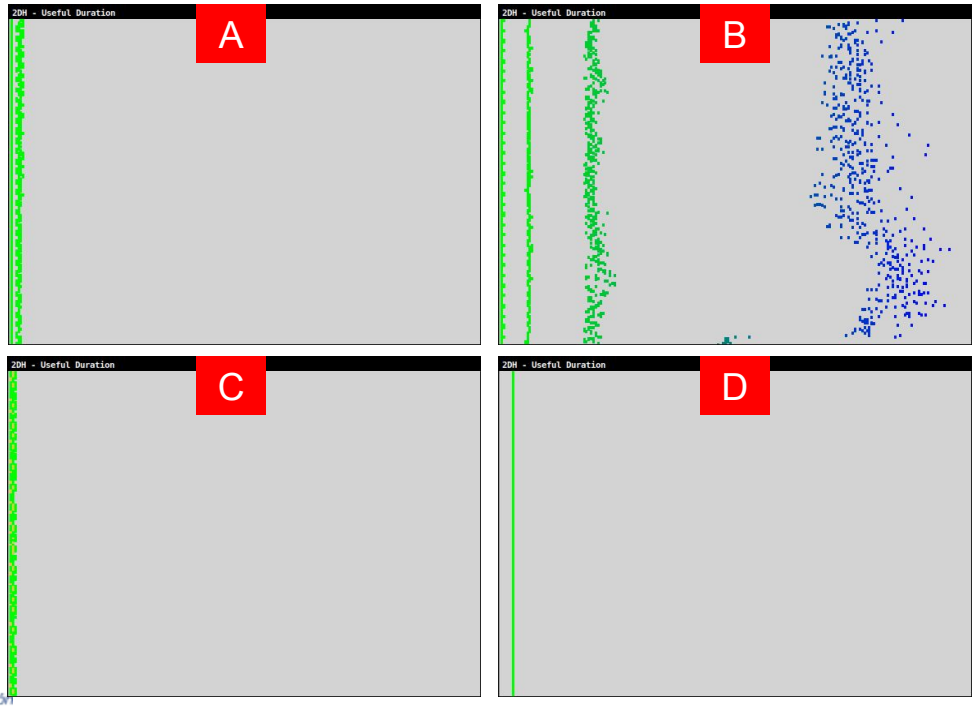
Performance model

- Computation and parallel efficiency factors for MPI only:
  - Good computation scalability and serialization efficiency
  - Not very good load balance neither transfer efficiency

Load balance

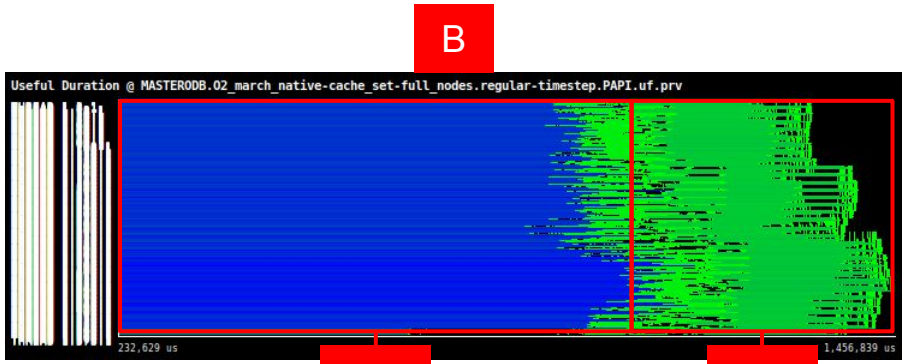
All phases are well balanced, except the grid-point computation



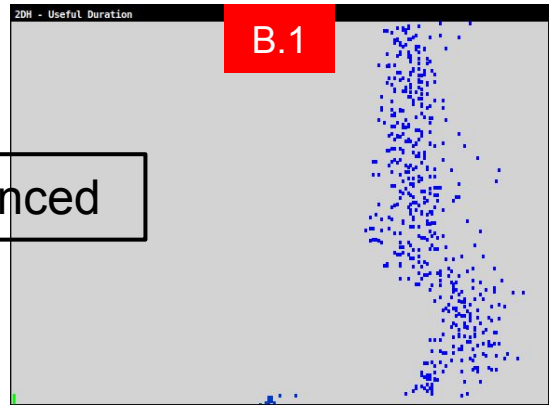


Load balance: grid-point comp. phase

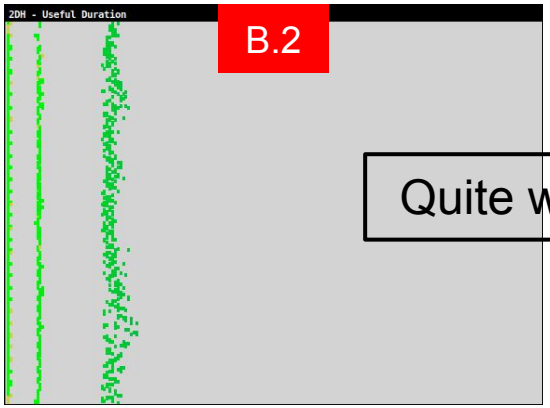
- B.1 part: CPG subroutine (cpg.F90) and inner subroutines
- B.2 part: Semi-Lagrangian interpolation that belongs to the CALL_SL subroutine (call_sl.F90) and inner subroutines



	Running
Total	46,923.83 %
Average	81.46 %
Maximum	97.81 %
Minimum	55.80 %
StDev	6.55 %
Avg/Max	0.83



Imbalanced

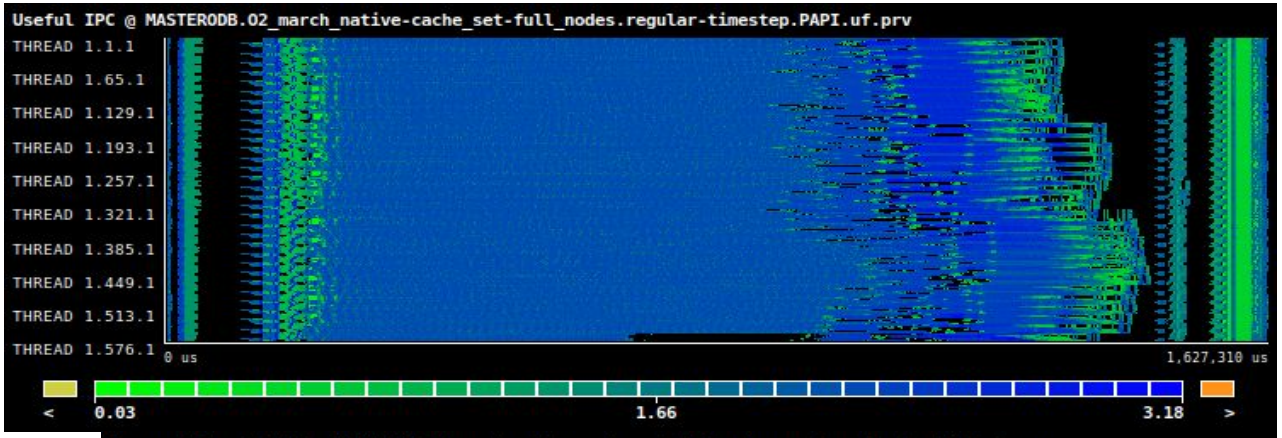


Quite well balanced

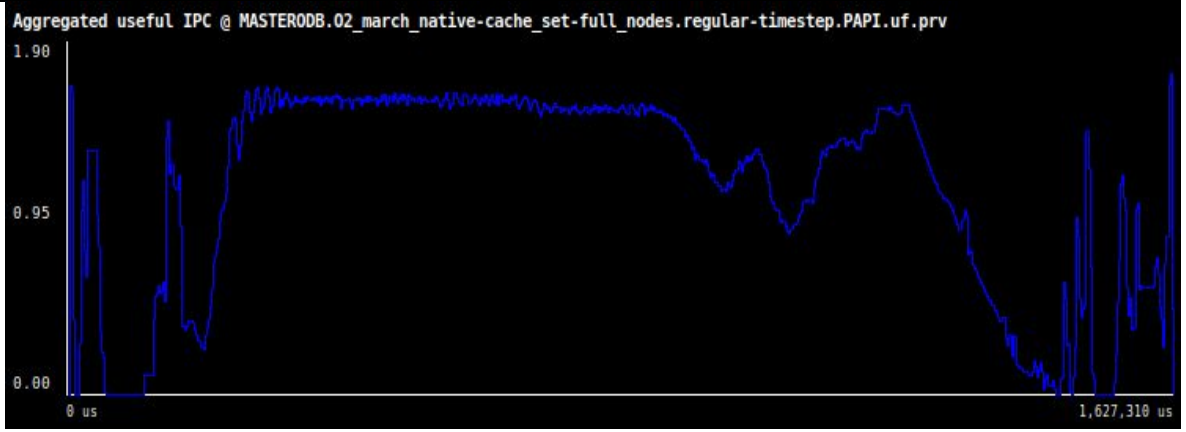
Instructions per Cycle (IPC)

In general, the IPC is above one in the computational parts, except in the spectral one that is quite low.

Useful IPC

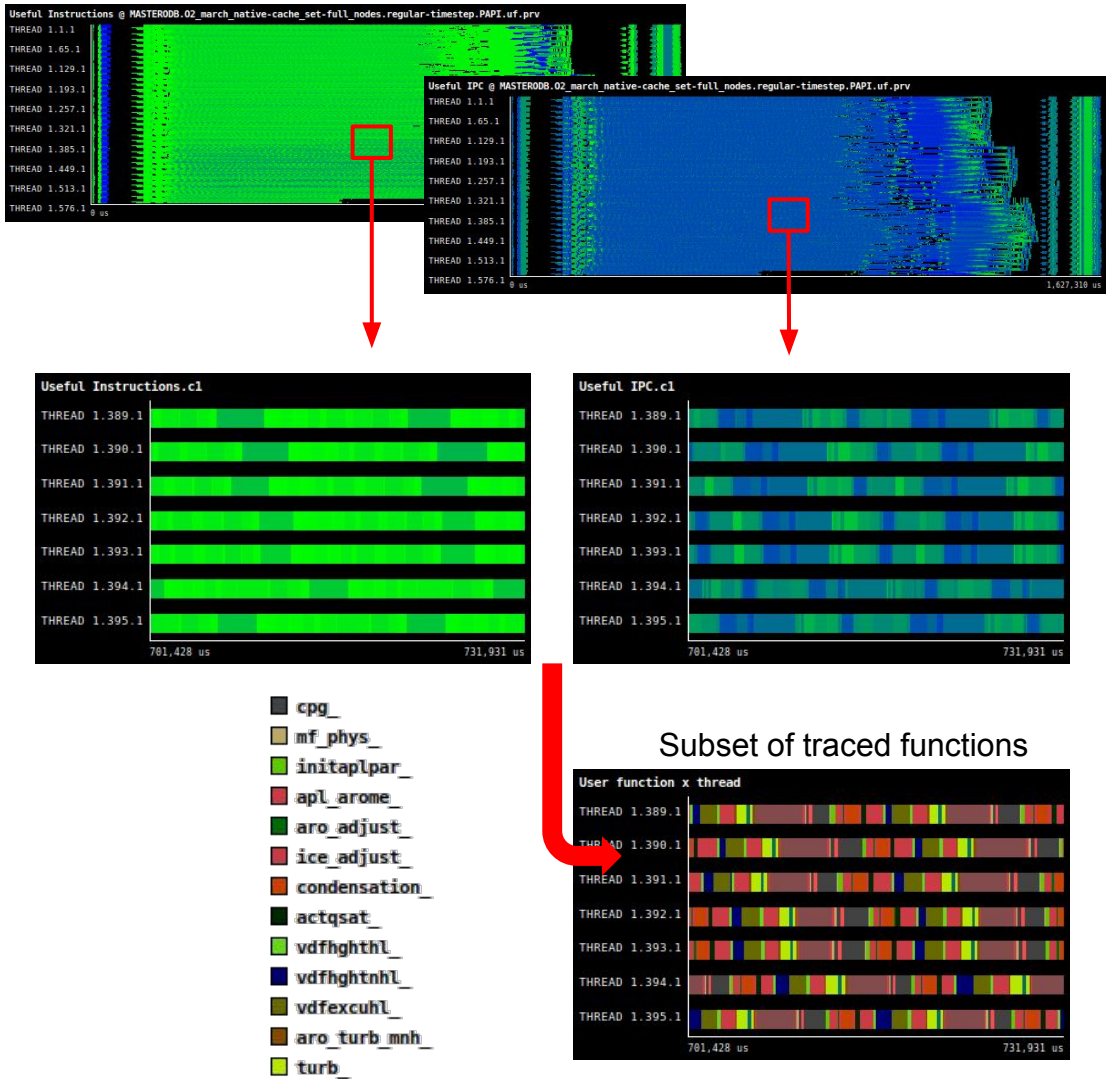


Aggregated IPC



IPC in detail

The IPC varies a lot depending on the piece of code that it is executing



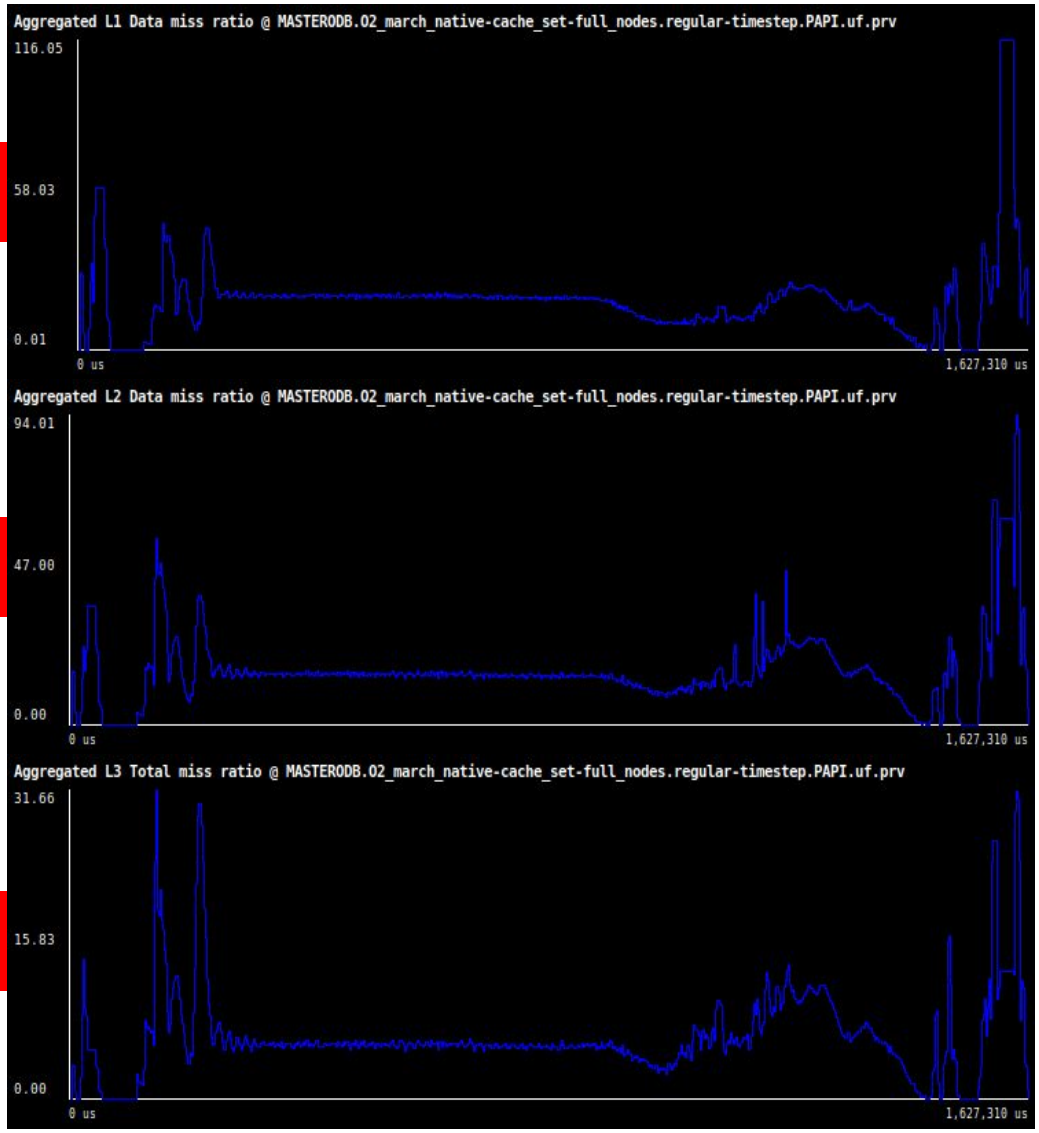
Cache misses

- The cache miss ratios in the grid-point part is constant and lower compared to other regions ⟹ high IPC
- The cache miss ratios in the spectral part is high ⟹ low IPC

L1

L2

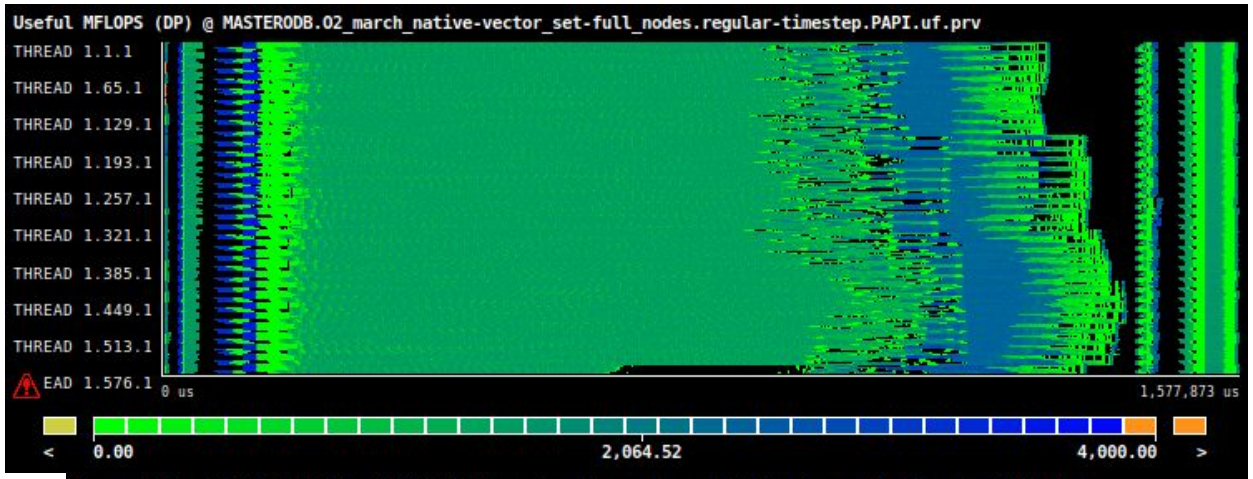
L3



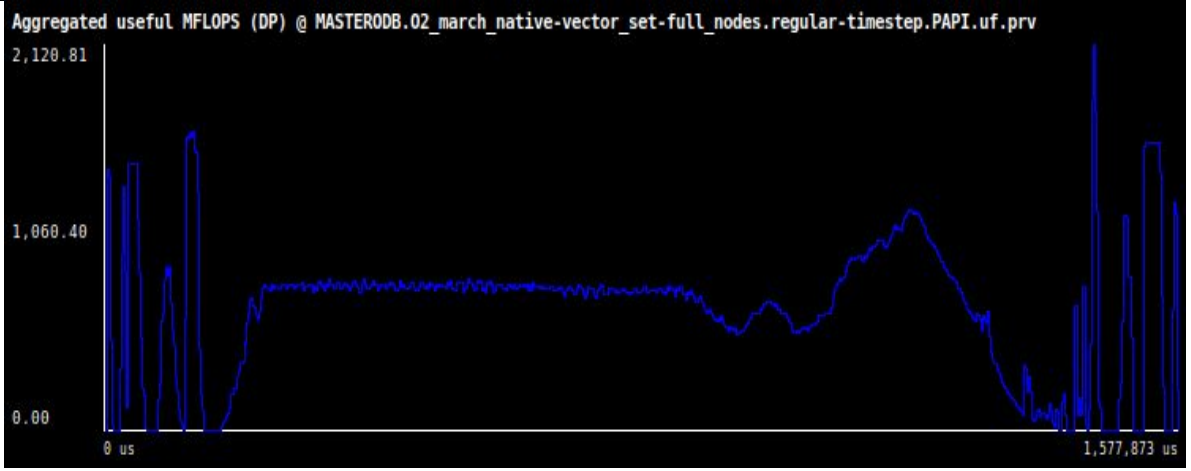
MFLOPS

In general, the MFLOPS do not surpass 1600 and for the grid-point computation phase is about 800

Useful MFLOPS



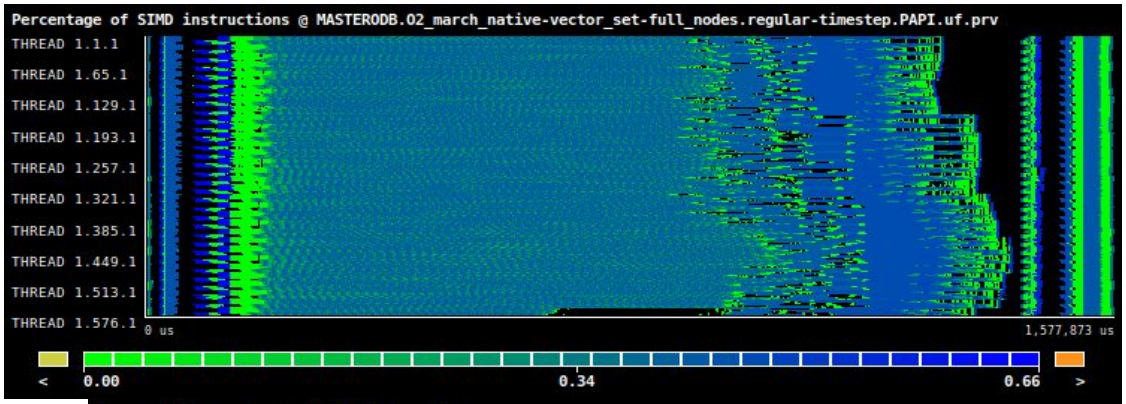
Aggregated MFLOPS



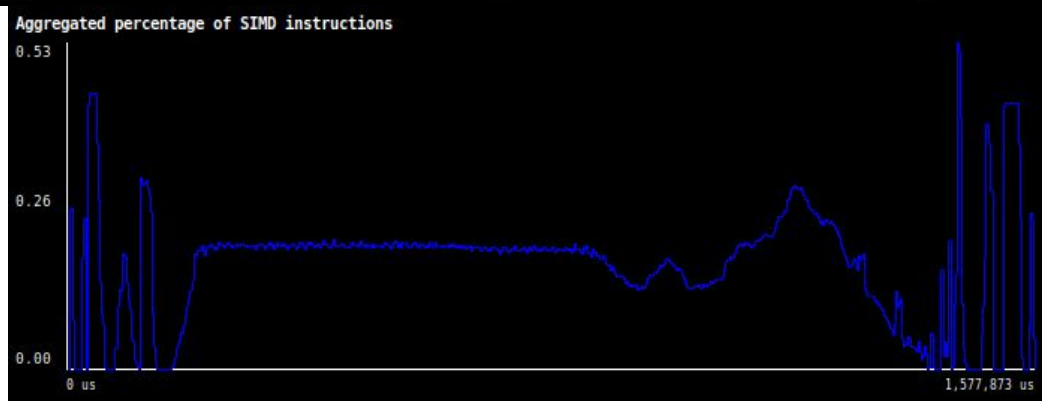
SIMD instructions

- Percentage of floating-point operation SIMD instructions of the total instructions: 20% for the grid-point phase and 43% for the spectral phase
- This percentage difference is the main reason in their MFLOPS difference
- In the grid-point phase there are more instructions than in the spectral phase that are not doing useful computation

Useful percentage



Aggregated percentage

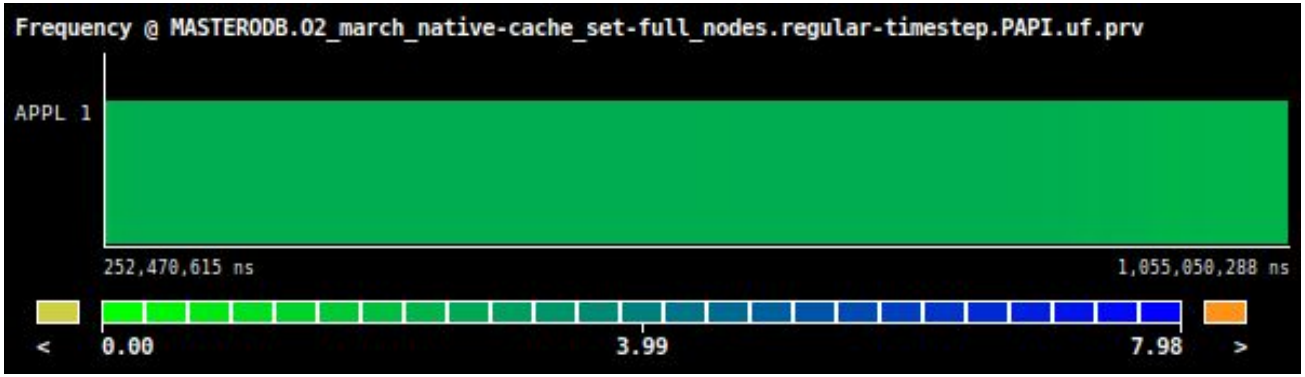


Sources of load imbalance

- CPU frequency slowdown: The system may decide to temporarily slow down the frequency because of thermal issues
- IPC slowdown: The number of instructions (IPC) achieved is very correlated to the performance of the application. There are many factors that may slow it down:
  - Increase in the number of cache misses.
  - Increase in the number of TLB misses.
  - Increase in the number of mispredicted branches.
  - Other types of stalls such as ROB, SB, etc.
- Different number of instructions: Because of the algorithm used or the parallelization strategy, the amount of instructions (workload) may not be balanced across the processes

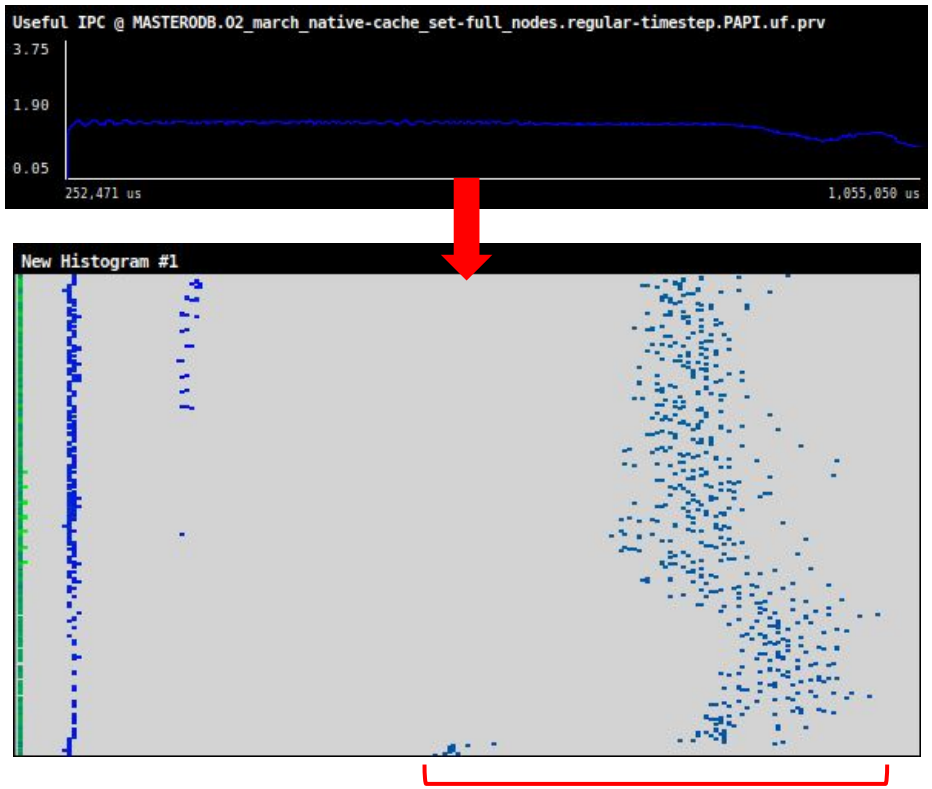
CPG load imbalance: CPU frequency?

The aggregated CPU frequency of the CPG part, which is constant along the time, it is **not** the cause of the load imbalance



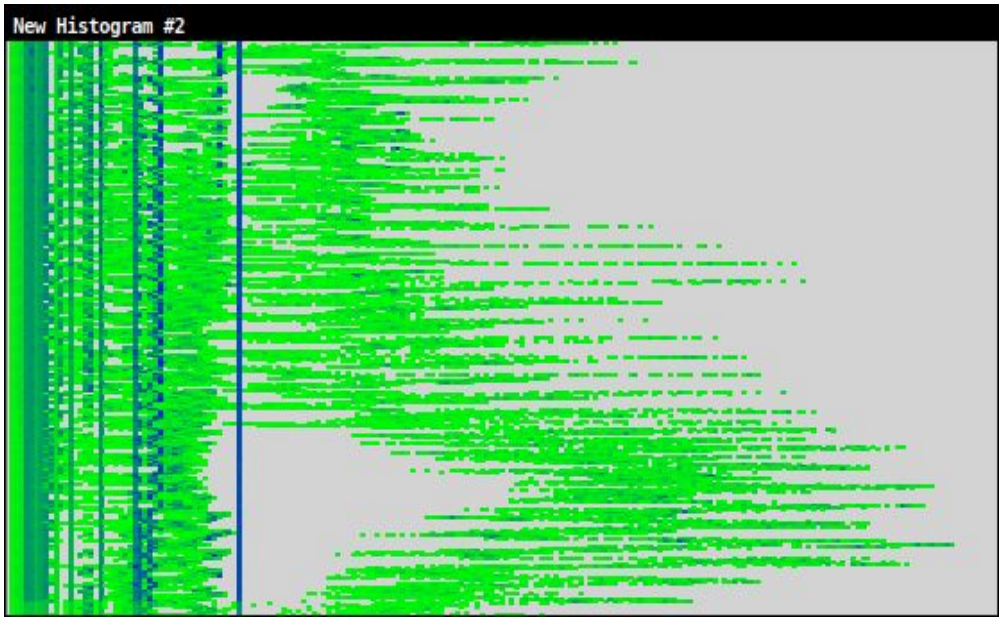
CPG load imbalance: IPC?

The IPC is kept across all processes, as there is no variance in the semantic scale (color) of the CPG part of the histogram, so it is **not** the cause of the load imbalance



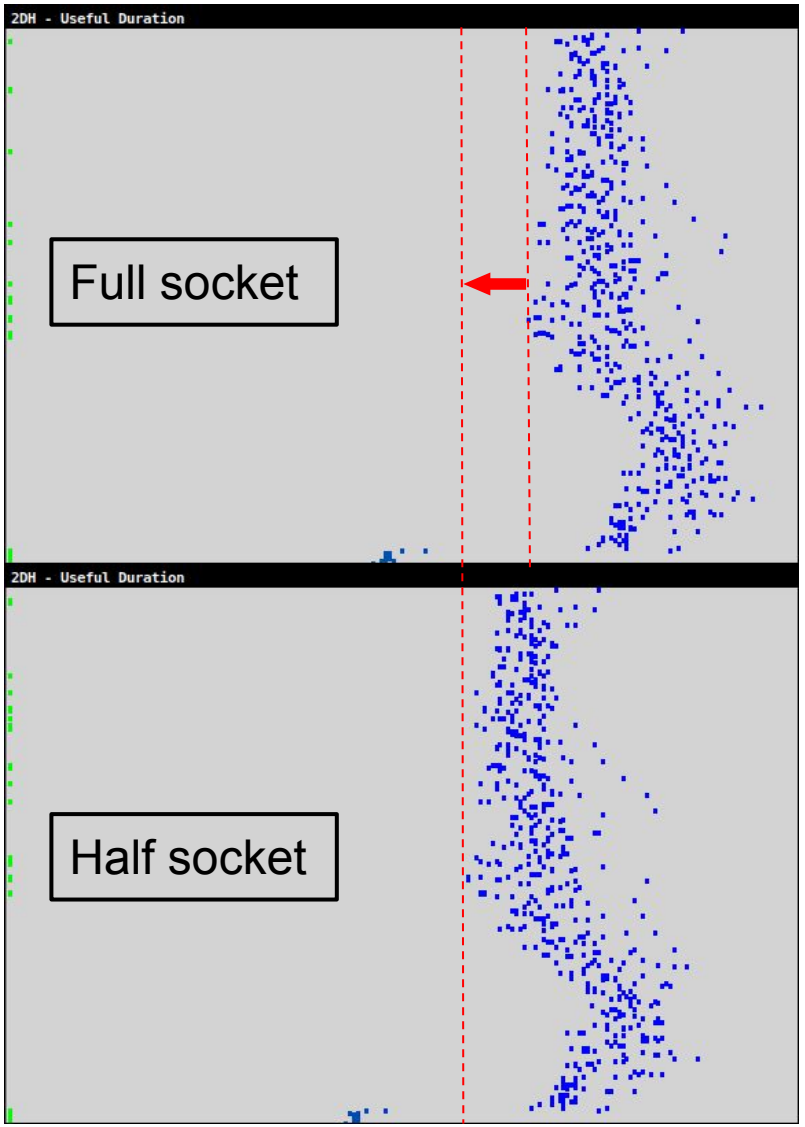
CPG load imbalance: instructions?

- The histogram indicates that the amount of work across processes is not the same, so this **is** the source of the load imbalance
- Two different sources: extension zone treatment and unknown



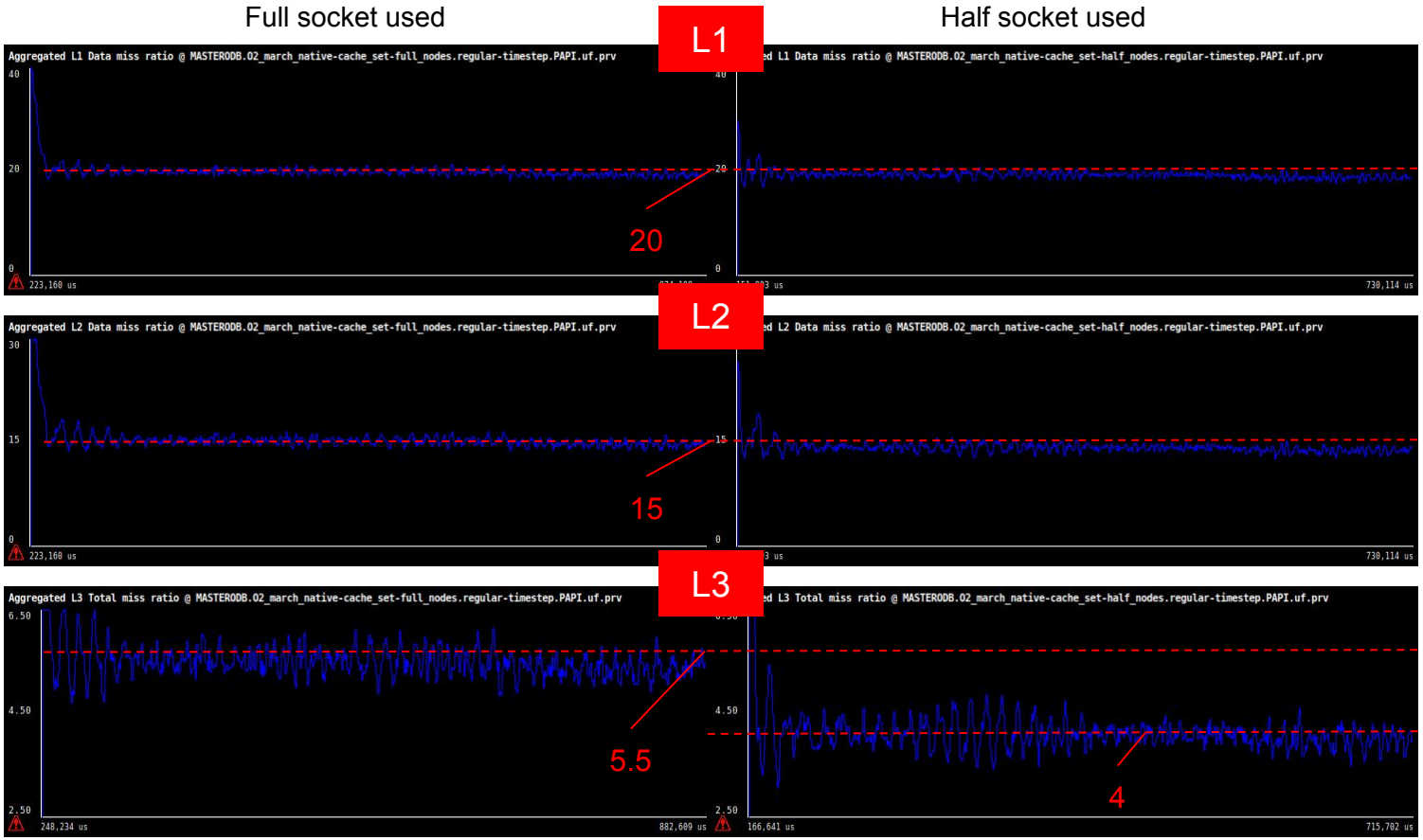
CPG part: compute or memory bound?

- Simple L3 cache pressure test: use only half of the cores per socket (18 instead of 36), but doubling the nodes
- There is a decrease in the duration of the bursts, which proves that L3 is a bottleneck, so in this context the application is **memory bound**.



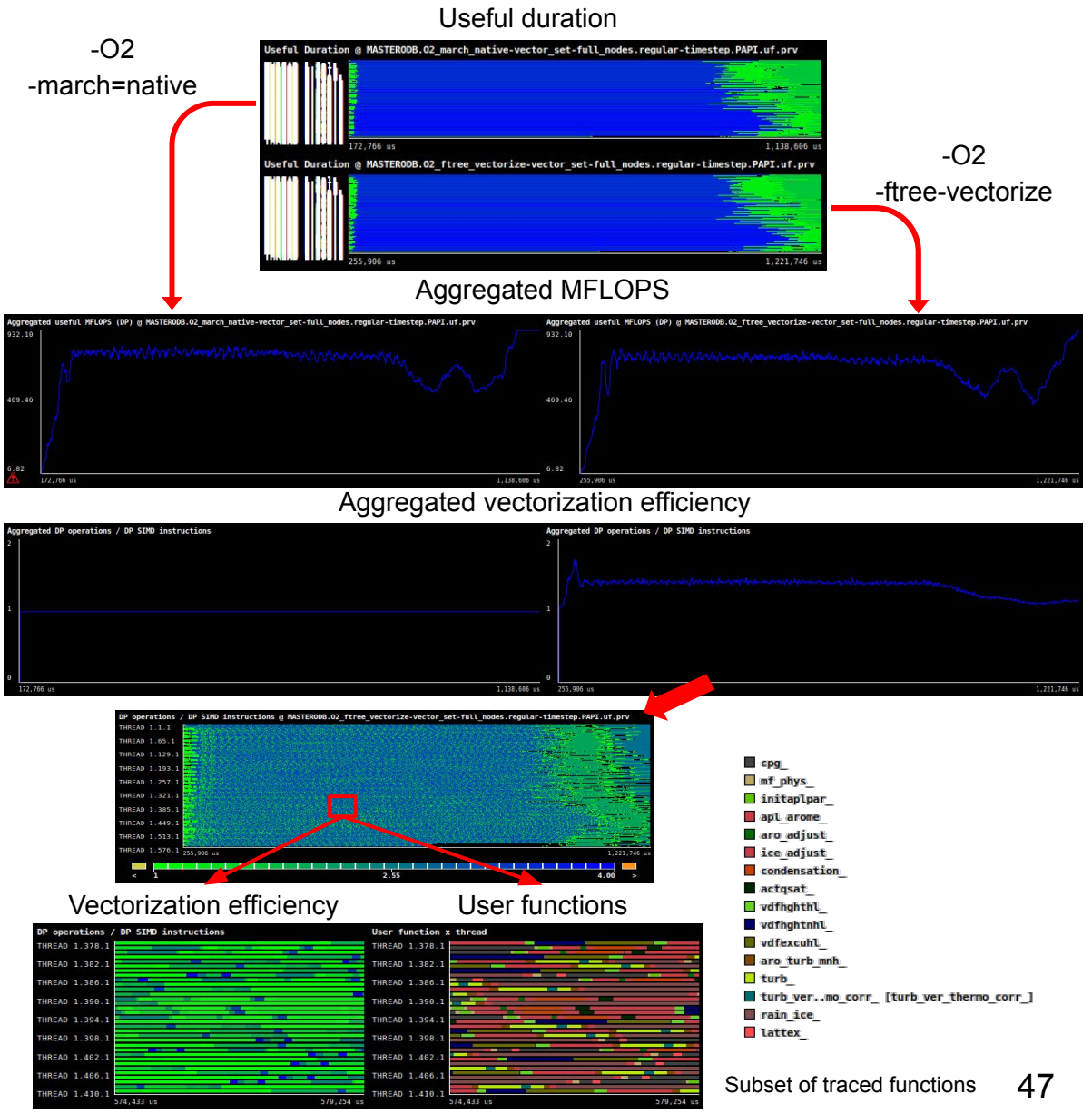
CPG part: compute or memory bound?

What happens? The L3 cache misses decrease when using half of the cores of the socket



CPG part: gcc auto-vectorization

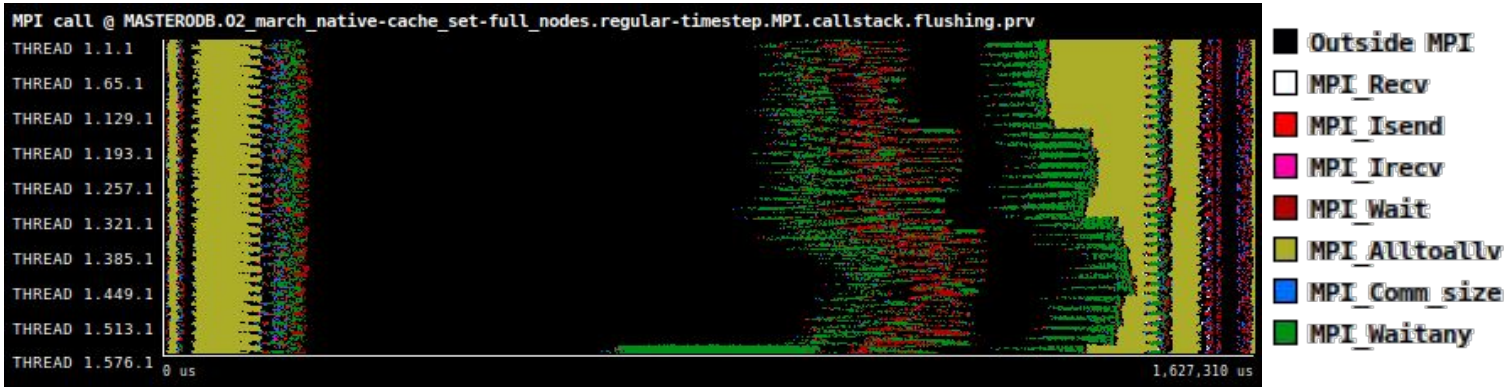
- The vectorization efficiency is around 1.40 for the vectorized code
- It really depends on the user function
- Difficult to understand why this poor auto-vectorization efficiency of gcc



Subset of traced functions

MPI calls and profile

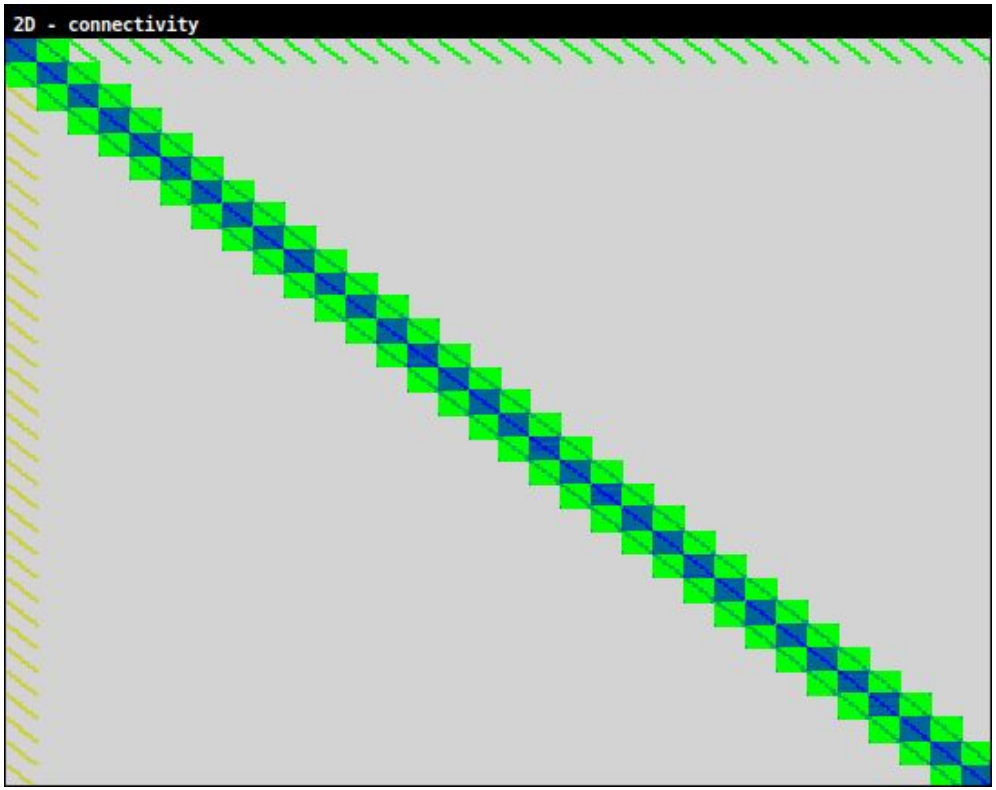
- Not many different types of MPI functions are used
- Only the MPI_Alltoallv and MPI_Waitany functions represent a significant amount of time with 14.65% and 9.29% respectively. In part, because of the CPG load imbalance



	Outside MPI	MPI_Recv	MPI_Isend	MPI_Irecv	MPI_Wait	MPI_Alltoallv	MPI_Comm_size	MPI_Waitany
Total	41,154.02 %	37.32 %	260.82 %	172.16 %	2,023.25 %	8,438.25 %	163.09 %	5,351.09 %
Average	71.45 %	0.07 %	0.45 %	0.30 %	3.51 %	14.65 %	0.28 %	9.29 %
Maximum	84.51 %	0.18 %	1.15 %	0.81 %	9.70 %	20.07 %	0.31 %	31.12 %
Minimum	51.79 %	0.00 %	0.13 %	0.12 %	0.49 %	8.59 %	0.23 %	1.31 %
StDev	4.91 %	0.04 %	0.18 %	0.14 %	1.79 %	3.07 %	0.01 %	4.33 %
Avg/Max	0.85	0.38	0.39	0.37	0.36	0.73	0.93	0.30

Point-to-point connectivity matrix

- It indicates who communicates with whom
- Almost all point-to-point communications are locally performed between MPI processes neighbours



Collective communications

- Four calls to MPI_Alltoallv each time step
- The most significant in terms of size and duration is the second one

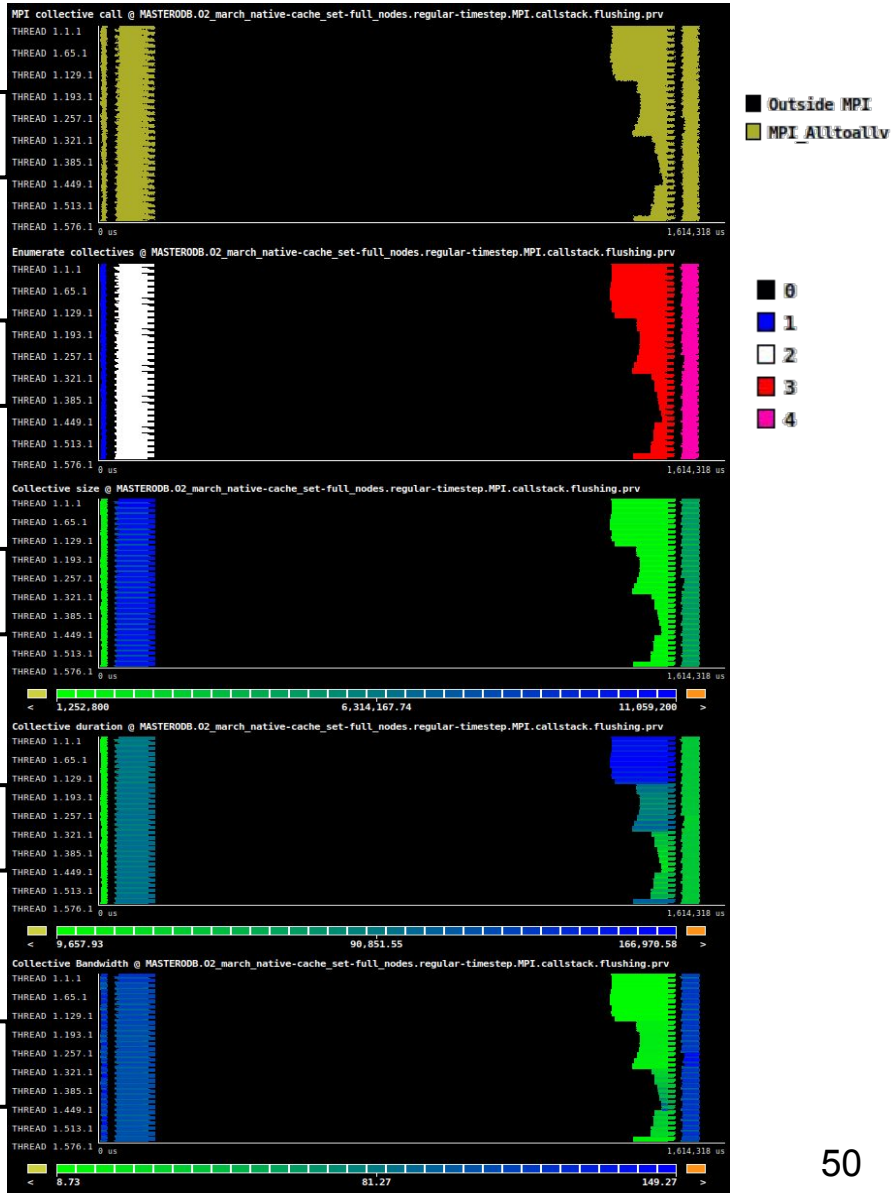
Call

Enumeration

Size

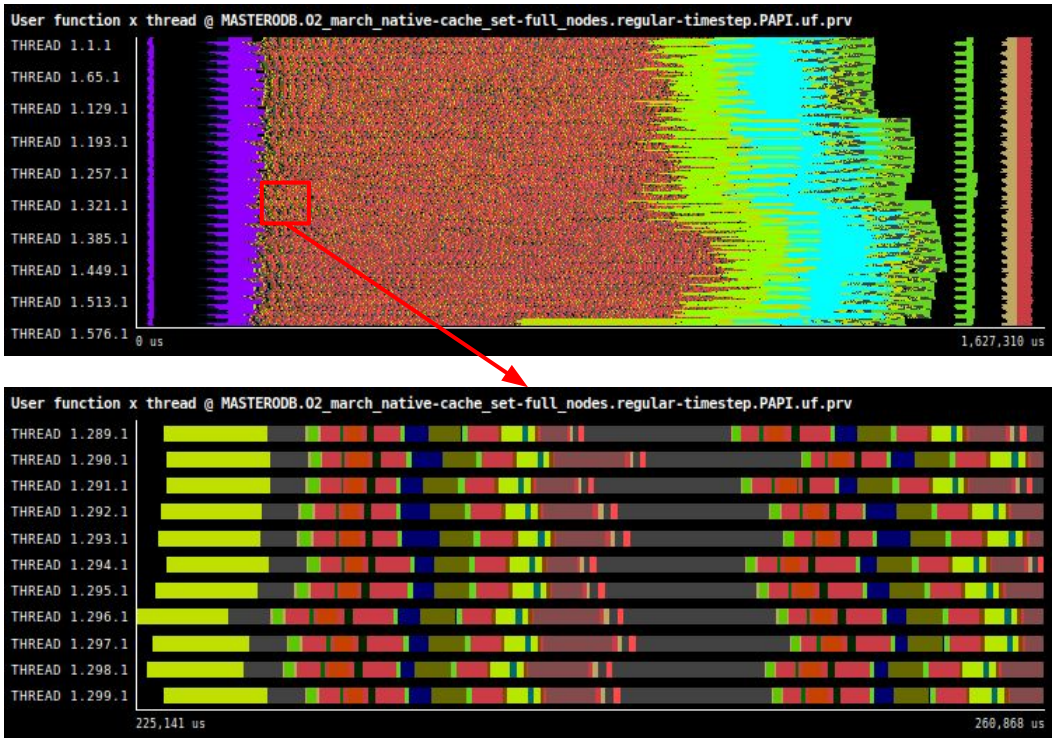
Duration

Bandwidth



Subset of profiled user functions

- The granularity of the calls is very fine
- Performance may be decreased because additional control instructions are executed



End
trgtol_comm
__fourie..urier_in [__fourier_in_mod_MOD_fourier_in]
trltog_comm
gp_model_
cpg_
mf_phys_
initaplpar_
apl_arome_
aro_adjust_
ice_adjust_
condensation_
actqsat_
vdfhghthl_
vdfhghtnhl_
vdfexcuhl_
aro_turb_mnh_
turb_
turb_ver..mo_corr_ [turb_ver_thermo_corr_]
rain_ice_
lattex_
elarmes_
lascaw_vintw_
lapineb_
laitri_
esc2r_
eseimpls_
ecoupl2_
trmtos_
espnhsi_

Recommendations

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Recommendations

- Building process: have two compilation files for the HARMONIE binary and the other tools
- Compilation flags:
  - Add the -march=native flag to use specific instructions of the target architecture
  - Remove the -ffast-math flag
  - Remove some (legacy) flags that might not be useful or necessary, such as -fno-second-underscore or -rdynamic
- OpenMP: if HARMONIE is only run with one OpenMP thread per MPI process, it would be useful to disable the OpenMP support

Recommendations (2)

- Job geometry for new platforms. On ECMWF cca these parameters should be defined:
  - -cc
  - -S
  - -ss
  - -d and EC_threads_per_task
  - -j
- The MPI master should run in a shared node to save SBU
- Explore alternative compilers that might provide a performance increase such as Intel
- Extension zone treatment: it would be interesting to find a workaround to mitigate the load imbalance

What's next?

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Follow-up ideas

- Whole time step:
  - Study in detail the point-to-point communications
  - Study the low efficiency of the OpenMP parallelization
- Grid-point computation phase, especially the CPG part:
  - The source of the load imbalance
  - NPROMA blocking scalability
  - Analyze other sources of CPU stalls
  - Determine if the fine granularity of user functions affects the performance
  - Determine performance metrics such as IPC or cache misses at a lower level, this is, per significant user function
  - Analyze the low auto-vectorization efficiency of gcc


Barcelona
Supercomputing
Center
Centro Nacional de Supercomputación
EXCELENCIA
SEVERO
OCHOA
Thank you
xavier.yepes@bsc.es